15-859(B) Machine Learning Theory

Homework # 5 **Due: March 29, 2010**

Groundrules: Same as before. You should work on the exercises by yourself but may work with others on the problems (just write down who you worked with). Also if you use material from outside sources, say where you got it.

Exercises:

1. [**PAC-learning OR-functions revisited**] Suppose the target function is a disjunction (OR-function) of r out of n boolean variables, for $r \ll n$. We know one good approach to learning in this case (which works even in the stringent on-line mistake-bound model) is to run the Winnow algorithm. Describe a different (simpler) algorithm for learning in the PAC model that finds a disjunction of $O(r \log m)$ variables that is consistent with the training data, where m is the number of training examples. Describe a variant of your algorithm that finds a disjunction of only $O(r \log(1/\epsilon))$ variables, with error at most $\epsilon/2$ on the training data. Note that by Occam + Chernoff bounds, the latter algorithm requires sample size at most $O(\frac{1}{\epsilon}((r \log 1/\epsilon) \log n + \log 1/\delta))$ to learn in the PAC model, so is just very slightly worse than that achieved via the mistake-bound to PAC conversion of the Winnow algorithm.

Problems:

2. [**SQ learning Decision Lists and Trees**] Give an algorithm to learn the class of *decision lists* in the SQ model (and argue correctness for your algorithm). Be clear about what specifically the queries χ are and the tolerances τ. Remember, you are not allowed to ask for conditional probabilities like $\Pr[A|B]$ but you can ask for $\Pr[A \wedge B]$.

   So, combined with your results from Homework 1, this gives an algorithm for learning decision trees of size s in the SQ model with queries and τ^{-1} on the order of $n^{O(\log s)}$, matching our SQ-dimension lower bounds.

3. [**Uniform distribution learning of DNF Formulas**] Give an algorithm to learn the class of *DNF formulas* having at most s terms over the *uniform distribution* on $\{0,1\}^n$, which has sample size polynomial in n and s, and running time $n^{O(log(s/\epsilon))}$. So, your algorithm matches the SQ-dimension lower bounds.

   Hint: Think of your algorithm for Exercise 1.

   Note: your solution to Problem 3 requires that data come from the uniform distribution, unlike your solution to Problem 2. The best algorithm known for learning polynomial-size DNF formulas over general distributions has running time roughly $2^{O(n^{1/3})}$.